ソフトウェア工学

02: ソフトウェア開発プロセス –
ウォーターフォール型開発と
反復型開発モデル

理工学部 経営システム工学科
庄司　裕子

今回のテーマ

⇨前回の復習
- ソフトウェア開発プロセス入門
  - プロダクトの品質とプロセスの生産性の向上
  - ウォーターフォール型モデル v.s. 反復型モデル
  - ヒューマン ファクター
  - 支援ツール／環境

2

復習: ソフトウェア工学の背景

- 1960年代末～1970年代
  （コンピュータの第3～3.5世代）
  - ハードウェアの急速な進歩により、ソフトウェア開発需要が急増
  - ソフトウェア開発技術が未成熟
  - ⇨ソフトウェア技術者の絶対不足が危惧（ソフトウェア危機）
- ソフトウェア危機の打開のため、ソフトウェアの系統的な開発方法論/技術/ツールの整備の必要性が認識された
  - ⇨ ソフトウェア工学の誕生（1970年代初頭）

3

復習: ソフトウェア工学の目的

- ソフトウェア/システム開発の両輪
  - プロダクト: 開発される物（アーティファクト）
  - プロセス: 開発の方法
- ソフトウェア工学の目指すもの
  - プロダクトの品質向上
  - プロセスの生産性向上

⇨「良いシステムを少ない工数で」開発できる方法論と技術の開発

4

今回のテーマ

✓前回の復習
⇨ソフトウェア開発プロセス入門
- プロダクトの品質とプロセスの生産性の向上
- ウォーターフォール型モデル v.s. 反復型モデル
- ヒューマン ファクター
- 支援ツール／環境

5

ソフトウェア開発プロセス入門

⇨プロダクトの品質とプロセスの生産性の向上
- ウォーターフォール型モデル v.s. 反復型モデル
- ヒューマン ファクター
- 支援ツール／環境

6

品質と生産性を低下させる要因

- 最初から「真の要求」を仕様化することにこだわる余り、
  - 要求定義フェーズから先に進まない
  - 誤った要求仕様に固定される
- プロジェクト終盤でプロダクトの致命的な欠陥が顕在化する
  - ⇨大幅なやり直しか、その場しのぎの泥縄式対応
- 既存の検証済アーティファクトを一からわざわざ作る
  - ⇨開発の遅れや品質低下のおそれ

7

リスクと無駄をなくす決め手は？

- 要求定義フェーズの泥沼化
  - ⇨ 実行可能なプロトタイプを手早く作成（ラピッド プロトタイピング）して、クライアントとデベロッパの相互理解を深め、真のクライアント要求を引き出しやすくする
- プロジェクト終盤での致命的欠陥の顕在化
  - ⇨ プロジェクト リスクの早期解消により、要求とのミスマッチを小さいレベルで潰していく
- 既存アーティファクトの新規作成
  - ⇨ 既存資産の共有/活用を図る再利用環境を構築

8

問題の複雑さとリスクへの対処

- 現在入手可能な情報だけで顧客の（真の）要求をすべて記述することは非現実的（要求そのものが時間的に変化することもある）

とは言え…

- 仕様化されている要求が完全でなければ、開発中のシステムには常にリスク（要求とのミスマッチ）がつきまとう

⇨ 要求と変更を管理し、情報が得られ次第、アーティファクトに適切な修正を加え、リスクを早めに解消していく

9

複雑さの取り扱い（Ⅰ）

- 「銀の弾丸」は存在しない（No silver bullets）
  - 何にでもそれ1つで解決できるような方法論を人間は求めるが、現在開発対象とされるような複雑なシステムでは、1つの方法論ですべてをカバーするのは不可能（少なくとも現時点では存在しない）。
  - ⇨利用可能な方法論を組み合わせる複合アプローチが必要
    - 問題を分割し、それぞれの部分問題をある1つの方法論でカバー

10

複雑さの取り扱い（Ⅱ）

- 「まず完全性ありき」という発想からの脱却
  - 最初に問題を完全に記述してからそれを解決するというトップダウン アプローチは、少なくとも大規模システムの場合は本質的に不可能
  - ⇨インクリメンタルな反復型開発が最も有望
    - 扱う要求を少しずつ増やしながら、分析→設計→実装→テストのサイクルを繰り返し、各サイクルでリスクの一部を解消する
    - 各サイクルのマイルストーンとして、実行可能なプロトタイプを作成し検証する

11

技術の進歩は複雑さを軽減するか？

- 技術が進歩すれば、それでできることが増えるので顧客の要件のレベルが上がり（あるいは新たなニーズが生まれ）、結局、解決すべき問題の複雑さが増大する場合がある
  - ⇨ 技術が進歩すれば開発すべきシステムの複雑さが増すという皮肉
- （例） セキュリティ
  - ⇨ ネットワーク（特にインターネット）がこれほど普及しなければ、セキュリティ問題がこれほど深刻化することはおそらくなかったであろう。

12

ソフトウェア開発プロセス入門

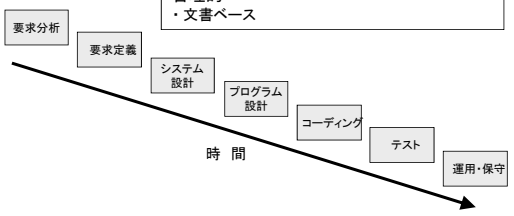
✓ プロダクトの品質とプロセスの生産性の向上
⇨ウォーターフォール型モデル v.s. 反復型モデル
❖ ヒューマン ファクター
❖ 支援ツール／環境

13

ウォーターフォール型開発モデル

しかし…
❖ 現実的でない

・トップダウン方式の問題解決手法
・上流から下流へとリニアなタスクの流れ
・上流で仕様が完全かつ明確に記述される必要がある
・仕様が最初に明確化できる場合には、最も合理的
・文書ベース



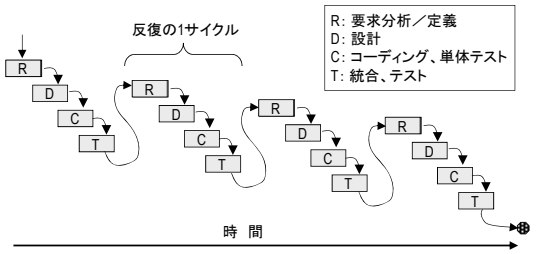
14

ウォーターフォール型モデルの欠点

❖ 最初に要求を完全かつ明確に記述できると仮定している
⇨そもそも、要求は変化する
❖ 設計の妥当性を紙上の検討だけで検証できると仮定している
⇨現実の問題に対してこれを可能にする技術はなく、人間による検証では見落しが発生する
⇨よく理解されている問題領域（ドメイン）以外は、このような仮定は非現実的

15

反復型開発モデル

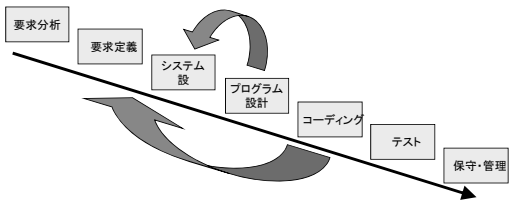


出典：フィリップ・クルーシュテン著『ラショナル統一プロセス入門』

16

偽の反復型開発モデル

❖ 以下のような反復は誤り
⇨ このような反復では問題の複雑さは緩和されず、リスクも解消されない。→ なかなか収束しない。



17

反復型開発モデルの特徴

❖ 対象とする問題の複雑さを段階的に緩和
⇨扱う要求を少しずつ増やしながら、R→D→C→Tのサイクルを繰り返し、各サイクルでリスクの一部を解消する
⇨各サイクルのマイルストーンとして、実行可能なプロトタイプを作成し検証する
❖ トップダウン方式（1つのサイクル内のR→D→C→T）とボトムアップ方式（次のサイクルへのステップアップ）の組み合わせ

18

反復型開発モデルの課題

- 最終的なプロダクトへの収束をどう保証するか
  - ⇨各反復が最初からの作り直しにならずに、インクリメンタルな繰り返しになるにはどうするか。
- 各反復で行う作業(扱う要求)と対処するリスクをどう選択するか
- 前の反復で見つかった重大な問題をどう解決するか

⇨個々の開発プロセス モデルごとにその手段が用意される必要がある(例: ラショナル統一プロセス)

19

ソフトウェア開発プロセス入門

✓ プロダクトの品質とプロセスの生産性の向上
✓ ウォーターフォール型モデル v.s. 反復型モデル
⇨ヒューマン ファクター
- 支援ツール／環境

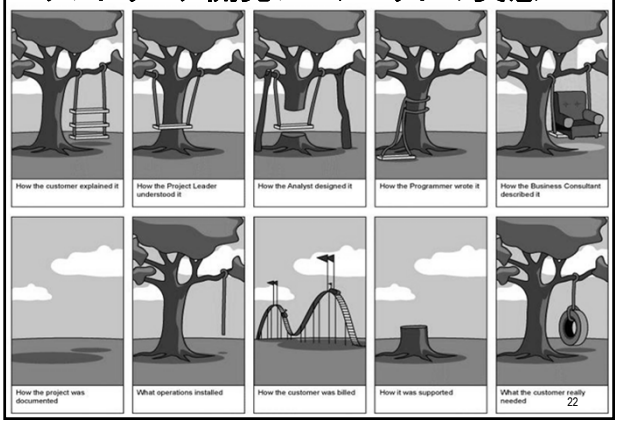
20

プロセスにおけるヒューマン ファクタ

- システム開発は純粋な技術論では済まず、さまざまな関係者の利害が複雑に絡み合うことが多い
  - ⇨ソーシャル スキルが物を言う。
- 特に、コミュニケーションが難しく、また重要

21

ソフトウェア開発プロジェクトの実態



22

ソフトウェア開発プロセス入門

✓ プロダクトの品質とプロセスの生産性の向上
✓ ウォーターフォール型モデル v.s. 反復型モデル
✓ ヒューマン ファクター
⇨支援ツール／環境

23

理想的なプロジェクト管理環境

- 要求管理ツール(要求と成果物のリンク)
- 変更管理ツール(成果物の一貫性を保つ)
- 構成管理ツール(種々のバージョンを管理)
- 統合開発環境(IDE)
- テスト ツール
- コミュニケーション支援ツール
  - メーリング リスト
  - ディスカッション グループ

24

統合開発環境（IDE）の主な機能

- UML ビジュアル モデリング
- フォワード/リバース エンジニアリング
- 入力支援機能付きプログラミング エディタ
- デバッガ
- ソースコード検査＆メトリクス測定
- リファクタリング
- 単体テスト
- ドキュメンテーション

25

まとめ：ソフトウェア開発プロセス

- ウォーターフォール型開発で対応できるプロジェクトは、すでに十分理解されている問題領域のみ
- 一般には、インクリメンタルな反復型開発が不可欠
  - 各サイクル内（R→D→C→T）はトップダウン
  - サイクル間のステップアップはボトムアップ
- 開発プロセスでは、技術論だけでなく、ヒューマンファクター（特に、コミュニケーション）も重要な要素

26

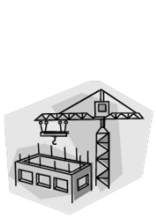
参考文献

- Philippe Kruchten,「The Rational Unified Process: An Introduction, Second Edition」, Addison-Wesley Pub Co, $34.99
- 「ラショナル統一プロセス入門　第2版」, ピアソン・エデュケーション 3,200円

27

考えてみよう

- プロセスの観点から見た、ソフトウェアの開発と建物の建設との類似点と相違点は何か



28